An Overview of Geometry Representation in Monte Carlo Codes

R.P. Kensek,[*] B.C. Franke,[*] T.W. Laub[*], L.J. Lorence,[*]

M. R. Martin,[*] S. Warren[†]

[*] Sandia National Laboratories, P.O. Box 5800, MS 1179, Albuquerque, NM 87185

[†] Kansas State University, Manhattan, KS 66506

Geometry representations in production Monte Carlo radiation transport codes used for linear-transport simulations are traditionally limited to combinatorial geometry (CG) topologies. While CG representations of input geometries are efficient to query, they are difficult to construct. In the Integrated-TIGER-Series (ITS) Monte Carlo code suite, a new approach for radiation transport geometry engines has been implemented that allows for Computer Aided Design (CAD), facetted approximations, and other geometry types to simultaneously define an input geometry. These techniques effectively allow the user to trade off problem setup time and computational time. For instance, radiation transport on CAD geometries can be 100 times slower than transport on comparable CG geometries. However, since linear-transport Monte Carlo can be solved with high parallel efficiency on massively-parallel computers, CAD representations of complex systems can be analyzed.

Introduction

Geometry representations in production Monte Carlo codes that are used for linear transport are traditionally limited to combinatorial geometry (CG) topologies (Nelson and Jenkins, 1988). In CG models, objects are described through Boolean operations on a set of simple primitives (volumes or surfaces). Such has been the case with Sandia's ITS or the Integrated TIGER Series code suite (Halbleib, et. al., 1992). While CG representations of input geometries are efficient to query, they are difficult to construct for model complexities required by engineers. Engineers typically create their designs using CAD software, where objects are described through a boundary representation (volumes are bounded by faces, which are bounded by edges, which are bounded by vertices). Hence, a path from CAD geometry into Monte Carlo codes is desirable to avoid the problem setup difficulties of re-creating a geometry in CG. Moreover, since engineers

frequently iterate on their designs, any path from CAD into a Monte Carlo code should not be cumbersome, but should accommodate rapid setup. An automated, or near automated, approach is desirable.

There are two general approaches to solving this problem: (1) Translate the model into combinatorial geometry, the format that Monte Carlo codes understand, or (2) Modify the Monte Carlo code to understand new geometry formats, into which CAD can be automatically translated. Many of the techniques that are being explored are associated with related issues of visualizing geometry and results from Monte Carlo calculations (Van Riper, 2004 and Schwarz, et. al., 2004).

Several research activities are underway that translate a geometry model into a CG representation. The TopAct code (Manson, 2004) has been written to convert CAD geometry into the surface and zone representations suitable for use in MCNP. In TopAct, a pre-processor developed by Technosoft, Inc. extracts analytical surface data from CAD data in STEP format (a standard recognized by most commercial CAD software). A mathematical algorithm is being developed to create semi-algebraic surface representations of CAD geometry suitable for MCNP (Tsige-Tamirat and Fischer, 2004). These approaches are still under active development, they are not yet generally available for users, and have not yet been adequately demonstrated for translating general CAD geometries into CG, particularly for spline surfaces.

Most CAD software includes built-in meshing utilities that can render CAD geometry into an unstructured mesh of tetrahedral or hexahedral elements. Hence, there have been various attempts to utilize meshed geometry with Monte Carlo. Brunner (Brunner et al. 2005) discusses a few codes using unstructured meshes especially designed to attempt to handle domain decomposition (needed to address possibly vast amounts of memory used) and the associated parallelization inefficiency. The medical-physics code PEREGRINE (Hartmann-Siantar et al. 1997) runs on a structured mesh ("voxels" appropriate for input based on patient CAT-scans) with special logic ("delta-scattering", which works better when there is not an enormous difference in mean-free-paths between materials used - such as tissue and bone) to mitigate the considerable cost involved with photons streaming across many mesh-cell boundaries. It is also possible to convert each cell of an unstructured mesh into CG zones to directly use with the CG-based Monte Carlo codes. However, our codes are not optimized in their data structures for representing such geometries and lack the special logic to efficiently accommodate streaming across many boundaries or domain decomposition - which would require a significant code restructure.

We have thus avoided meshing the entire problem which brings with it greatly increased memory (which may lead to complications such as handling domain decomposition for parallelization) and unnecessary (for radiation transport) internal boundary crossings. Even when we are required to calculate spatial distributions in parts (hence need a mesh-like structure), we utilize what we call "subzone" structures which are boundaries only seen for purposes of tallying spatial distributions. The subzone boundaries are not seen as the particles are tracked nor do they create any approximation to the surfaces of the zones (as a mesh typically does on curved surfaces). Since they are regular, in a local body-based coordinate system (cylindrical, spherical or Cartesian), they

require minimal memory to be fully described. Furthermore, they are usually restricted to regions of interest, so entire geometries are rarely subzoned for such tallies.

There are significant issues associated with use of CAD geometries (meshed or not) for particle transport, especially worrisome for designs that are rapidly evolving. We will refer to them as Design to Analysis (D2A) issues:

The CAD geometry has to be “clean”: "air-tight" (else may need to be "healed") without significant overlaps. The size of acceptable "small" overlaps is clearly application dependent.

Material identifiers need to be assigned to CAD zones and need to be conveyed to the representation (which may be lost if translated from one version of CAD to another, or meshed - if they were assigned to the original CAD zones in the first place).

For complex models, very numerous but trivial CAD details (i.e. those which may not be needed for a particular analysis) may make the model unusable (i.e. the model is so large it cannot be automatically integrity checked). This can be aggravated for meshing, where even a small number of such features may give rise to a needlessly large (for radiation transport) mesh. The analyst may need to iterate with the designer to create the appropriate level of detail (and avoid troublesome details for meshing).

For very complex systems, subassemblies are often designed separately, then "fit" together (again, this is the overlap problem, which in this case cannot be detected by looking at individual subassemblies). With meshes, care must be taken for curved boundaries between two materials where a possibly different effective facetization on either side of the boundary may again aggravate the overlap problem.

Our approach (to make use of CAD geometry designs) is to directly transport on CAD geometries (Franke et al., 2001 and Tautges et al., 2004). This has been implemented in Version 5.0 of ITS (Franke et. al., 2004) and will be discussed in the next sections. The D2A issues listed are still of concern with this approach, but the difficulties peculiar to meshing are avoided. In particular, we will discuss how overlaps in the CAD geometry are detected and treated in ITS. However, CAD-based transport does have a significant drawback. It is significantly slower than CG-based transport on comparable geometries (by factors of 10-100).

Another approach that is being investigated is transport on facetted representations of the geometry (Martin and Warren, 2004 and Jordan, 2004). While this approach has similar D2A drawbacks (for radiation transport) of a volumetric mesh, it does have reduced memory requirements (by avoiding the internal mesh boundaries) and may allow for Monte Carlo transport without domain decomposition. The generation of facetted surfaces is a common technique in visualization. Hence, many utilities exist to automatically generate a facetted geometry.

We allow, or plan to allow CG, CAD, and facet representations in ITS. Moreover, we are enabling our transport algorithms for models that are hybrid combinations of any of these representations. For example, a hybrid CAD-facet representation can be used to reduce computational cost associated with transporting particles across spline surfaces.

Transport across such surfaces can be factors of 10-100 times slower than transport across the CAD boundary representations that are not splines. Hence, it is beneficial to allow the replacement of spline surfaces with facetted representations in the Monte Carlo geometry. Visualization utilities can automatically generate models with selected facetization that is needed for such analysis.

ITS Version 5.0

In ITS Version 5.0, a representation-independent query engine has been developed that allows for Computer Aided Design (CAD) and facetted approximation to simultaneously define an input geometry. ITS is now used routinely for transport on CAD geometries, and the efficiency and accuracy of facetted and mesh approximations are being investigated. ITS5 has the ability to track particles on CAD geometry in the ACIS (Spatial, 2005) format. Using this feature frees the analyst from the chore of rebuilding geometries already available in a CAD format. This can significantly shorten the time required to perform an analysis on a complicated system. The use of CAD geometry representations trades geometry setup time for computational cost. That is, creating the geometric model for a calculation requires much less analyst time while performing the calculation requires more processor-hours. Particle tracking on CAD geometries requires more CPU time due to the added computation necessary to handle the CAD geometry representation (volumes bounded by finite surfaces, instead of volumes described by Boolean combinations of simple primitives), especially if the CAD model uses more general surfaces which are non-analytic. In the end, the turn around time of an analysis is reduced significantly if a CAD model of the geometry already exists. Most popular CAD formats can be converted to the ACIS format.

Improvements in CAD Geometry Particle Tracking Efficiency

There are two aspects of tracking on CAD geometry models which make it less efficient than tracking on CG models: (1) the intersections of rays with faces (trimmed surfaces) are more computationally expensive since there is no simple Boolean logic underlying the zone description, and (2) the space between objects is typically not defined. The first item we more or less cannot avoid - although we can try to minimize the calls to CAD libraries. One way to deal with the second item is to require that space to be defined - i.e. create a zone as the complement of all the other objects (Tautges et al. 2004), though this may result in a very complicated description (tracking on which may become computationally expensive). Our approach has been to allow the existence of such a zone (we shall call this the "special void" zone), conceptually defined as not being within any of the CAD parts, along with an efficiency mechanism (in particular a type of overlaying grid). An efficiency mechanism is needed for two related geometry queries: (1) determining location within this special void region without interrogating every zone in the problem, and (2) determining which surface will be intersected next from within this special void zone without interrogating every surface in the problem.

Our first efficiency grid simply divided space up uniformly along Cartesian axes to form cells. These cells were populated with any zone that was within or intersected the

cell. This scheme limited the number of zones (and by their association, surfaces) to be examined for queries related to the special void zone. For example, location within the special void region was determined by only examining all zones within one cell. Voxel-walk logic was included for cases when the "next surface" to be intersected was not located within the same cell as the particle. However, for some applications with small regions of large numbers of surfaces, a uniform grid was not very effective.

As part of a restructuring of the geometry tracking engine (Martin and Warren, 2004), a new type of efficiency grid was proposed and is being investigated. This new grid is a kind of Binary Space Partitioning (BSP) tree. BSP's are a set of spatial data structures that subdivide space to achieve greater efficiency in ray firing applications. The type of BSP tree being investigated is a K-dimensional binary tree, or KD tree, where all of the spatial subdividing planes are aligned with the axes. One of the advantages of the KD tree is that it adapts to the model by refining itself near points of high geometric detail. Another advantage is that subdivisions are selected based on a cost function that can be altered to fit a particular class of problems. Figure 1 illustrates the spatial partitioning represented by a KD tree developed for a complex geometry model.

Figure 1. The Spatial Partitioning Represented By a KD Tree Developed for the Complex Model.

The cells of the KD tree are populated with faces which are within or intersect the cell. The two queries for the special void zone are more related, since location within the special void zone is determined by examining the "next surface" which would be intersected, then deciding if the particle is inside or outside the associated zone to that surface. If it is outside, then it is inside the special void zone.

The Problem with Overlapping Geometry

Figure 2 illustrates a simple overlap (labeled C) of zone B and zone D. Zone A is the special void zone. A particle being tracked is located at the black dot in zone B, moving in direction indicated by the arrow. For a given application, it may be that region C is so small it would not affect the results if that region were considered as part of zone B or part of zone D. With our simple uniform spatial grid, the code identified region C as part of zone B in this case. It would identify it as part of zone D for a particle in zone D moving towards zone B.

Figure 2. The overlap of Zone B with Zone D is indicated as region C.

However, with the newer KD tree and the alternate logic which determines location within the special void zone, a problem occurs. Without special logic, it would identify the overlap region C as part of zone D - which is not the problem. However, having made that association, the code determines the black dot (in zone B) is outside zone D, which is the definition of being inside the special void region. The danger is that it decides the black dot is inside the special void region, and will stream to region C. To avoid this potentially disastrous situation, whenever the code decides a point is located inside the special void region (hence outside some zone), it performs the query a second time, but now ignoring any surfaces associated with the zone it identified as being outside. This is not so terribly inefficient, since the code has stored previous intersections for surfaces it may have already interrogated. Hence, with this special logic, the code will associate the overlap region C with zone B (since zone D is being ignored for this second query), and will properly identify the black dot as being within zone B. The code recognizes this as an overlap, and prints out a diagnostic as such (essentially once per pair of discovered overlapping zones). While the code robustly handles such simple overlaps, multiple overlaps could still potentially cause trouble (i.e. erroneous results), so users should strive to obtain clean CAD geometries if the code identifies overlaps.

Use of Facet-Based and Other Geometry Representations

One of the main advantages of Martin's geometry tracking engine is its modular nature. The modular nature of the engine allows the addition of other geometry representations besides CG and CAD (in ACIS format) without a wholesale rewrite of ITS. One of the alternate geometry representations being investigated is facet-based geometry. As mentioned above, tracking on CAD geometry is much slower than on CG geometry. This is especially true if the CAD geometry contains spline surfaces, which are common in CAD geometry. An advantage of facet-based geometries is that there are no spline surfaces—there are only plane surfaces. However, many plane surfaces may be needed to adequately represent any given surface. A facet-based geometry ability is already present in ITS5 and investigations are currently underway are to determine if facet-based surface geometry will be efficient enough and at the same time accurate enough to replace some or all CAD bodies in a model. Figure 3 illustrates a test of the facet-based geometry particle tracking logic in ITS. The facetted bowl has 1616 faces. The CAD and CG geometries produced identical results for the particular calculation. The facet-based geometry calculation produced nearly identical results and as the resolution of the facets was refined the results continued to improve in comparison with the other three. The effect of splines on tracking time is clear.

Geometry FeatureTime (ms/history)CAD Splines64.3CAD No Splines0.65Facets0.22 CG 0.0066**Figure 3.** Comparison of Monte Carlo run times on a bowl model CAD geometry with and without splines, facet-based geometry, and CG geometry.

Another advantage of Martin's geometry tracking engine is that particles can be tracked on different geometry representations in a single calculation. In particular, the tracking engine resolves the issue of gaps and overlaps associated with stitching disparate geometries together. If there is a simple overlap, the code chooses one of the parts when there is an overlap. Gaps are treated as void.

This means that ITS5 can track particles through a geometry that consists of CG zones, CAD zones, and facetted-body zones. Thus it may be possible to optimize a given model for the most efficient representation or combination of representations.

CAD Subzoning

For extracting spatial differential data, ITS5 has extended automated subzoning capabilities for the combinatorial geometry (CG) and CAD versions. The 3D code can now handle more single-body zone types, multiple-body zones, and can overlay any of the subzone structures on any given zone. This is called non-conformal subzoning as the subzone entity does not conform to the shape of the zone. Subzones that are outside of the zone are identified and no tallying is performed in those subzones. All subzoning schemes also have automatic subzone volume calculations. One exception to subzone volume calculation is for non-conformal subzoning on CG geometry. In this case those subzones that are identified as outside the zone (based on the centroid of the subzone) are given a volume of zero. Some of these subzones actually have some volume inside the zone. A stochastic volume calculation can be used to determine the fractional volumes of subzones on the boundaries of zones and the subzone volumes can be specified in the input to override automatically calculated volumes. Subzoning of zones in CAD geometries is of the overlay type by default, see Figure 4 for an example. In CAD-based transport, the CAD routines calculate volumes for all subzones using geometric intersection logic.

Subzoning is available for any geometry brought into ITS (hence it is currently available for facet geometry). But special logic is needed for precise volume calculation with subzone overlays. That special logic is in place for CAD geometry and is being developed for facet geometry.

Figure 4. Regular subzone overlay of CAD geometry.

Summary

Sandia's production Monte Carlo code for electron-photon transport, ITS, has been adapted for flexible utilization of CG, CAD, and facetted geometries. No one method is superior in all circumstances. To take advantage of the strengths and weaknesses of each approach, ITS can also transport on hybrid models that combine any or all of these methods.

Acknowledgements

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States National Nuclear Security Administration and the Department of Energy under Contract DE-AC04-94AL85000.

References

Brunner, T.A., Urbatsch, T.J., Evans, T.E., and Gentile, N.A., "Comparison of Four Parallel Algorithms for Domain Decomposed Implicit Monte Carlo", this conference and submitted to Journal of Computational Physics, 2005.

Franke, B.C., Kensek, R.P., Schriner, H.K., Lorence, L.J., Gelbard, F. and Warren, S., "Adjoint Charge Deposition and CAD Transport in ITS," M&C2001 Salt Lake City, Utah (2001).

Franke, B.C., Kensek, R.P., and Laub, T.W., "ITS Version 5.0: The Integrated TIGER Series of Coupled Electron/Photon Monte Carlo Transport Codes with CAD Geometry," SAND2004-5172, Sandia National Laboratories (2005).

Halbleib, J. A., Kensek, R. P., Mehlhorn, T. A., Valdez, G. D., Seltzer, S. M., and Berger, M. J., "ITS Version 3.0: The Integrated TIGER Series of Coupled Electron/Photon Monte Carlo Transport Codes," Technical Report SAND91-1634, Sandia National Laboratories (1992).

Hartmann-Siantar,C.L., Bergstrom, P.M., Chandler, W.P., Chase, L., Cox, L.J., Daly, T.P., Garrett, D., Hornstein, S.M., House, R.K., Moses, E.I., Patterson, R.W., Rathkopf, J.A., Schach von Wittenau, A., "Lawrence Livermore National Laboratory's PEREGRINE Project", Proc. 12th International Conference on the Use of Computers in Radiation Therapy, Leavitt, D.D. and Starkschall, G., eds., Medical Physics Publishing, Madison, Wisc., (1997).

Jordan, T.M., "Using CAD Files in the Novice Code," *Trans. Am. Nucl. Soc*. **91**, 187 (2004).

Martin, M.R. and Warren, S., "Multiple Geometry Representations in Monte Carlo Radiation Transport, " *Trans. Am. Nucl. Soc*. **91**, 183 (2004).

Manson, S.J., "TopAct: Automated Translation from CAD to Combinatorial Geometry,"

Trans. Am. Nucl. Soc. **91**, 181 (2004).

Nelson, W.R. and Jenkins, T.M., "Geometry Methods and Packages," pp. 385-405, in "Monte Carlo Transport of Electrons and Photons," Ed: T. M. Jenkins, W. R. Nelson, and A. Rindi (Plenum, 1988).

Schwarz, R., Lewis, R.R., and Roetman, V.E., "mcnpviz: A Program for the Interactive Display of an MCNP Geometry," *Trans. Am. Nucl. Soc*. **91**, 176 (2004).

Spatial, a subdivision of Dessault Systems, "3D ACIS Modeler," http://www.spatial.com/components/acis/ (2005).

Tautges, T.J., Wang, M. and Henderson, D.L., "CAD-Based Monte Carlo Transport Using MCNPX and CGM", *Trans. Am. Nucl. Soc*. **91**, 185 (2004).

Tsige-Tamirat, H. and Fischer, U., "CAD Based Geometry Generation for Monte Carlo Particle Transport Codes, *Trans. Am. Nucl. Soc*. **91**, 179 (2004).

Van Riper, K.A., "White Rock Science Tools for Geometry Modeling," *Trans. Am. Nucl. Soc*. **91**, 183 (2004).